One-way Jumping Finite Automata (New contact points of algebraic systems, logics, languages, and computer sciences)

AUTHOR(S):

千川原, 寛之; 山村, 明弘

数理解析研究所講究録
第1964巻 2015年 3-14

One-way Jumping Finite Automata

HIROYUKI CHIGAHARA, SZILÁRD ZSOLT FAZEKAS, AKIHIRO YAMAMURA

Department of Computer Science and Engineering, Akita University

1 Introduction

Most classical computer science methods process information in a continuous way. For instance, in finite automata theory [4, 7], the read head of finite automata starts at the beginning of the input word, it moves in left-to-right direction, and it reads symbol-by-symbol. Jumping finite automata [8] were proposed as automata process information in a discontinuous way. Jumping finite automata can jump over a part of the input word after reading a symbol and continue processing from there. Once a symbol in the input word is read, it cannot be re-read again later during computation of it. The read head starts anywhere in the input word to check the input word whether it is accepted or not, and it can move in either direction; that is, jumping finite automata are non-deterministic in terms of reading input word. In [8], it is shown that the language class accepted by jumping finite automata does not completely include finite language, regular language, and so on. They establish several results concerning jumping finite automata regarding commonly studied areas of automata theory, such as decidability and closure properties.

We propose (right and left) one-way jumping finite automata which are one of variants of jumping finite automata. Right one-way jumping finite automata move similar to jumping finite automata except for that the read head moves deterministically left to right starting from the left most letter in the input and when it moves to the end of the input word, then it returns to the beginning of the input word and continues the computation. Furthermore, if there are some symbols to read by the rule, then jumping finite automata reads the symbol which is the nearest to read head in the right direction of it. Left one-way jumping finite automata are defined similarly.

We obtain a pumping lemma for the language accepted by (right or left) one-way jumping finite automata as well. The lemma is useful to decide whether or not a certain language is accepted by one-way jumping finite automata. In addition, we show relationship between the family of languages accepted by (right or left) jumping finite automata and classical language families. We also show that the language accepted by right one-way jumping finite automata is different form the language accepted by left one-way jumping finite automata. We report results obtained but omit proofs. The details will be published elsewhere.

2 Preliminaries

An *alphabet* Σ is a finite, non-empty set and its elements are called *letters*. A word is a sequence of letters and the set of all words formed by concatenating elements of Σ is Σ^*. The *empty word*, i.e., the word containing no letters is ε.

A *(nondeterministic) finite automaton*, a FA for short, is a quintuple $M = (Q, \Sigma, R, s, F)$, where Q is the finite set of states, Σ is the finite input alphabet, $\Sigma \cap Q = \emptyset$, $R \subseteq Q \times \{\Sigma \cup \varepsilon\} \times Q$, $s \in Q$ is the start state, and $F \subseteq Q$ is the set of final states. Elements of R are referred to as rules of M and we write $py \to q \in R$ instead of $(p, y, q) \in R$. A configuration of M is a string in $Q \times \Sigma^*$. M is an ε-free FA if $py \to q \in R$ implies $|y| = 1$. M is a *deterministic finite automaton*, a DFA for short, if (1) it is an ε-free FA and (2) for each $p \in Q$ and each $a \in \Sigma$, there is no more than one $q \in Q$ such that $pa \to q \in R$. A FA or DFA makes a transition from configuration pw to configuration qw' if $w = aw'$ and $pa \to q \in R$, where $p, q \in Q$, $w, w' \in \Sigma^*$ and $a \in \Sigma \cup \{\varepsilon\}$. We denote this by $pw \Rightarrow qw'$ and the reflexive and transitive closure of the relation $\Rightarrow$ by $\Rightarrow^*$. A word w is *accepted* by a FA (or DFA) M if there exists $f \in F$, such that $sw \Rightarrow^* f$. Then, the language accepted by M is $L(M) = \{w \in \Sigma^* \mid \exists f \in F : sw \Rightarrow^* f\}$

A *jumping finite automaton* defined in [8], a JFA for short, is a quintuple $M = (Q, \Sigma, R, s, F)$, where Q, Σ, R, s and F are the same as in the case of (nondeterministic) finite automata. Therefore, JFA are "based" on FA, but transitions, and therefore the sets of accepted words, are different. A configuration of M is any string in $\Sigma^* \times Q \times \Sigma^*$. The binary jumping relation, symbolically denoted by $\curvearrowright$, over $\Sigma^* \times Q \times \Sigma^*$, is defined as follows. Let x, z, x', z' be strings in Σ^* such that $xz = x'z'$ and $py \to q \in R$; then, M makes a jump from $xpyz$ to $x'qz'$, symbolically written as $xpyz \curvearrowright x'qz'$. Note that in every step as above, the JFA deletes the letter it just read (y). In the standard manner, we extend $\curvearrowright$ to $\curvearrowright^m$, where $m \geq 0$. Let $\curvearrowright^+$ and $\curvearrowright^*$

denote the transitive closure of $\curvearrowright$ and the transitive-reflexive closure of $\curvearrowright$, respectively. The language accepted by M, denoted by $L(M)$, is defined as $L(M) = \{uv \mid u, v \in \Sigma^*, usv \curvearrowright^* f, f \in F\}$, i.e., the language accepted by M is the set of all words, such that starting from the initial state s the JFA M reads and deletes all letters of the input word. Let w be a string in Σ^*. We say that M accepts w if $w \in L(M)$ and that M rejects w otherwise.

The *cardinality* of Q, denoted by $|Q|$, is the number of elements of Q. We extend this notation to words $w \in \Sigma^*$, where $|w|$ is the *length* of the word w, the number of all occurrences of all letters of Σ in w. Further extending the notation, $|w|_a$ denotes the number of occurrences of the letter a in w. As usual, a *permutation* of n elements is a bijection $\sigma : \{1, 2, \ldots, n\} \to \{1, 2, \ldots, n\}$, which can be represented by $\sigma = \begin{pmatrix} 1 & 2 & 3 & \cdots & n-1 & n \\ \sigma(1) & \sigma(2) & \sigma(3) & \cdots & \sigma(n-1) & \sigma(n) \end{pmatrix}$, or assuming that the upper row is always in increasing order, by $\begin{pmatrix} \sigma(1) & \sigma(2) & \sigma(3) & \cdots & \sigma(n-1) & \sigma(n) \end{pmatrix}$.

A *permutation of a word* $w = a_1 a_2 \ldots a_n$ is a rearrangement of letters of w, namely, some word $w_\sigma = a_{\sigma(1)} a_{\sigma(2)} \ldots a_{\sigma(n)}$, where σ is a permutation of n elements. The set of all permutations of w is denoted by $Perm(w)$. For any language L, the set $\bigcup_{w \in L} Perm(w)$ is denoted by $Perm(L)$.

It is shown in [8] that the class of languages accepted by JFA and regular languages are incomparable, and JFA is closed under union, intersection and complement, respectively. It is easily shown that if a language L is accepted by a JFA then L must coincide with $Perm(L)$. Therefore, there is no JFA accepting the language $\{a\}^*\{b\}^*$.

Throughout the rest of this paper, the language families under discussion are denoted in the followings; **FIN**, **REG**, and **CF** stand for the families of finite languages, regular languages, and context-free languages, respectively. **JFA** denote the families of languages accepted by JFAs.

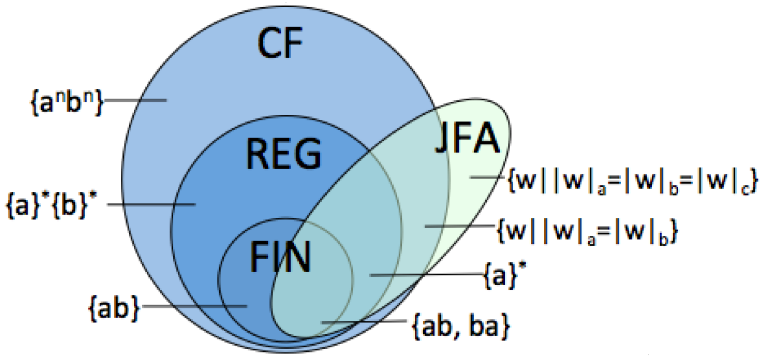

Figure 1. Relation between **JFA** and well-known language families

3 One-way jumping finite automata

In this section, we define variants of jumping finite automata, that is, a left and right one-way jumping finite automaton. A right one-way jumping finite automaton M is based on a deterministic finite automaton, but it does not read the input string in a symbol-by-symbol left-to-right way. The read head of M starts at the leftmost letter of the input word and it moves rightward. M can jump over a part of the word after reading a symbol. If the read head of M reaches the right end of the word, then it continues from the left end, again. The image of how to read the input word is Figure 2. The beginning and the end of input word are joined like a circle. The read head moves to the clockwise rotation and searches the symbol to read.

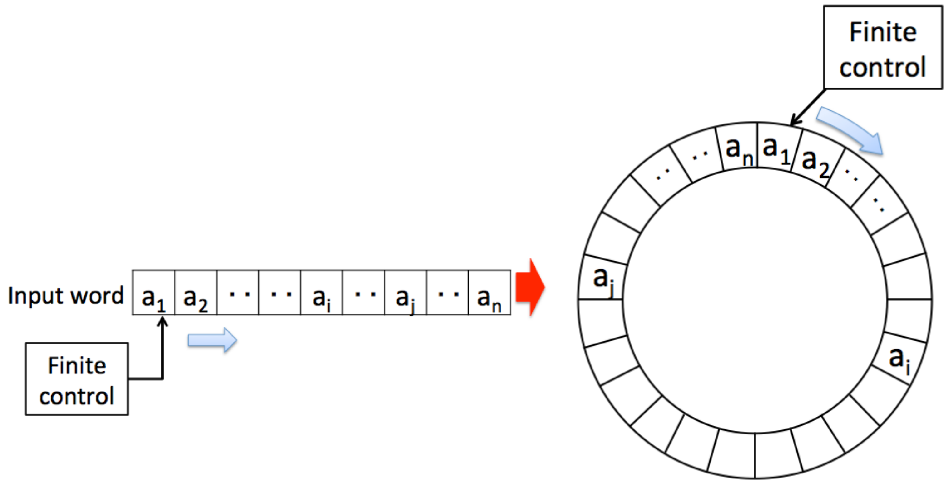

Figure 2. The image of how to read the input word

Likewise, the read head of a left one-way jumping finite automaton N starts at the right end of the input word and it moves leftward, continuing the computation from the right end upon reaching the left end. In the following formal definitions transitions are different from previous automata models.

A *right one-way jumping finite automaton*, ROWJFA for short, is a quintuple $M = (Q, \Sigma, R, s, F)$, where Q, Σ, R, s and F are defined as in a DFA. By analogy with a DFA, members of R are referred to as rules of M and we write $pa \to q \in R$ instead of $(p, a, q) \in R$. A configuration of M is any string in $Q\Sigma^*$.

The *right one-way jumping relation*, symbolically denoted by $\circlearrowright$, over $Q\Sigma^*$, is defined as follows. Suppose that x and y belong to Σ^*, a belongs to Σ, p and q are states in Q and $pa \to q \in R$. Then the ROWJFA M makes a jump from the configuration $pxay$ to the configuration qyx, symbolically written as

$$pxay \circlearrowright qyx$$

if x belongs to $\{\Sigma \setminus \Sigma_p\}^*$ where $\Sigma_p = \{b \in \Sigma \mid (p, b, q) \in R \text{ for some } q \in Q\}$.

In the standard manner, we extend $\circlearrowleft$ to $\circlearrowleft^m$, where $m \geq 0$. Let $\circlearrowleft^*$ and $\circlearrowleft^+$ denote the transitive-reflexive closure and the transitive closure of $\circlearrowleft$, respectively.

The language accepted by the ROWJFA M, denoted by $L(M)$, is defined to be

$$L(M) = \{w \in \Sigma^* \mid sw \circlearrowleft^* f, f \in F\}$$

We say that M accepts a string w in Σ^* if w belongs to $L(M)$, and M rejects w otherwise.

A left one-way jumping finite automaton, a LOWJFA for short, is a quintuple $M = (Q, \Sigma, R, s, F)$, where Q, Σ, s and F are defined as in an DFA. Members of R are referred to as rules of M and instead of $(q, a, p) \in R$, we write $q \leftarrow ap \in R$. A configuration of M is any string in Σ^*Q. The binary jumping relation, symbolically denoted by $\circlearrowleft$, over Σ^*Q, is defined as follows. Let x,$y \in \Sigma^*$, $a \in \Sigma$, p,$q \in Q$ and $q \leftarrow ap \in R$; then, M makes a jump from $yaxp$ to xyq, symbolically written as

$$yaxp \circlearrowleft xyq$$

if x belongs to $\{\Sigma \setminus \Sigma_p\}^*$ where $\Sigma_p = \{b \in \Sigma \mid (q, b, p) \in R \text{ for some } q \in Q\}$.

In the standard manner, we extend $\circlearrowleft$ to $\circlearrowleft^m$, where $m \geq 0$. Let $\circlearrowleft^*$ denote the transitive-reflexive closure of $\circlearrowleft$.

The language accepted by M, denoted by $L(M)$, is defined as

$$L(M) = \{w \in \Sigma^* \mid ws \circlearrowleft^* f, f \in F\}$$

Let $w \in \Sigma^*$. We say that M accepts w if and only if $w \in L(M)$, and rejects it otherwise.

Example 1 Let M_1 be a ROWJFA given by

$$M_1 = (\{q_0, q_1\}, \{a, b\}, R, q_0, \{q_0\}),$$

where R consists of the rules $q_0a \rightarrow q_1$ and $q_1b \rightarrow q_0$.

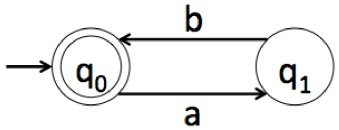

Figure 3. The state diagram for M_1

Starting from q_0, M_1 has to read some a and some b entering again the start (and also the final) state q_0. All these occurrences of a and b can appear anywhere in the input word. Therefore, M_1 accepts the non-regular context-free language $L_1 = \{w \in \{a,b\}^* \mid |w|_a = |w|_b\}$.

Example 2 Let M_2 be a ROWJFA given by

$$M_2 = (\{q_0, q_1, \ldots, q_{n-1}\}, \{a_1, a_2, \ldots, a_n\}, R, q_0, \{q_0\}),$$

where R consists of the rules $q_i a_{i+1} \to q_{i+1}$, for all i with $0 \leq i < n-1$, and $q_{n-1}a_n \to q_0$.

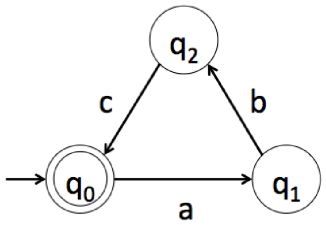

Figure 4. The state diagram for M_2

Starting from q_0, M_2 has to read an a_1, then an a_2 and so on, finally reading an a_n and entering again the start (and also the final) state q_0. All these occurrences of a_i, $1 \leq i \leq n$, can appear anywhere in the input word. Therefore, the accepted language is $L_2 = \{w \in \{a_1, a_2, \ldots, a_n\}^* \mid |w|_{a_1} = |w|_{a_2} = \cdots = |w|_{a_n}\}$, which is not context-free for any $n > 2$.

Example 3 Let M_3 be a ROWJFA given by

$$M_3 = (\{q_0, q_1\}, \{a, b\}, R, q_0, \{q_0, q_1\}),$$

where R consists of the rules $q_0 a \to q_0$, $q_0 b \to q_1$ and $q_1 b \to q_1$.

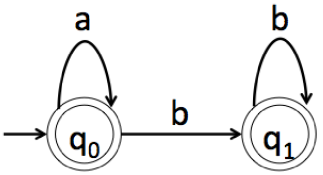

Figure 5. The state diagram for M_3

Starting from q_0, M_3 has to read an arbitrary number of symbols a. Then, it can read an arbitrary number of symbols b. Therefore, M_3 accepts a language $L_3 = \{a\}^*\{b\}^*$ which cannot be accepted by any JFA(see [8]).

Let $M = (Q, \Sigma, R, s, F)$ be a ROWJFA. Let there exist a sequence of configurations $\chi_0, \cdots, \chi_n$ for some $n \geq 1$ such that $\chi_{i-1} \circlearrowright \chi_i$ $[q_{i-1}a_{\sigma(i)} \to q_i]$ where $q_i \in Q$ and $a_i \in \Sigma$ for all $i = 1, \cdots, n$; that is,

$$\begin{array}{rl} \chi_0 \circlearrowright \chi_1 & \left[q_0 a_{\sigma(1)} \to q_1\right] \\ \circlearrowright \chi_2 & \left[q_1 a_{\sigma(2)} \to q_2\right] \\ \vdots & \\ \circlearrowright \chi_n & \left[q_{n-1} a_{\sigma(n)} \to q_n\right] \end{array}$$

Then, M makes n moves from χ_0 to χ_n according to $r_1 \cdots r_n$, written as

$$\chi_0 \circlearrowright^n \chi_n \quad \left[(q_0 a_{\sigma(1)} \to q_1) \cdots (q_{n-1} a_{\sigma(n)} \to q_n)\right]$$

The definition above formalizes the intuitive notion of computing with a finite automaton, which, if not completely defined, jumps to the nearest symbol to the right, which it can process.

4 Jumping finite automata

We answer open problem 17 of Meduna and Zemek [8]. They ask a necessary and sufficient condition for a language to be accepted by a JFA. The authors state that a necessary condition is that the language is closed under taking all permutations of its words, i.e., a language L is accepted by some JFA only if $L = perm(L)$ ([8], Theorem 13). By a simple argument, we show that we only need to add the condition that the Parikh image of the language is semilinear, in order to get a complete characterization.

A linear set $L = (c; p_1, \ldots, p_r)$ is a subset

$$\{x \mid x = c + \sum_{i=1}^{r} k_i p_i \text{ for some nonnegative integers } k_i, i = 1, \ldots, r\}$$

of $\mathbb{N}^r$, where c, $p_1, \ldots, p_r$ are elements of $\mathbb{N}^r$. A subset of $\mathbb{N}^r$ is called semilinear if it is a finite union of linear sets. The Parikh-image of a word w over an alphabet $\Sigma = \{a_1, \ldots, a_n\}$ is the vector $\Psi(w) = (|w|_{a_1}, \ldots, |w|_{a_n}) \in \mathbb{N}^{|\Sigma|}$. The Parikh image of a language L is

$$\Psi(L) = \{\Psi(w) \mid w \in L\} \subseteq \mathbb{N}^{|\Sigma|}.$$

We say that a language is linear or semilinear if its Parikh image is linear or semilinear, respectively. It is obtained in [9] that a regular language is semilinear and that for any semilinear language L, there exists a regular language L' such that $\Psi(L) = \Psi(L')$, where $\Psi(L)$ is the Parikh image of L. We obtain a necessary and sufficient condition as follow.

Theorem 1. *For any language $L \subseteq \Sigma^*$ the following statements are equivalent:*

- *there exists a JFA M, such that $L(M) = L$.*
- *L is commutative and semilinear.*

5 Languages accepted by right one-way jumping finite automata

Throughout the rest of this paper,we denote the families of languages accepted by right one-way JFA and left one-way JFA by **ROWJ** and **LOWJ**, respectively.

5.1 Basic properties

We obtain the following result.

Theorem 2. *The classes* **ROWJ** *and* **LOWJ** *are incomparable.*

Theorem 3. *Let* $M = (Q, \Sigma, R, s, F)$ *be a ROWJFA such that* $|\Sigma| = 1$. *Then,* $L(M)$ *is regular language.*

Corollary 4. *There is no ROWJFA that accepts* $\{a^p |\ p$ *is a prime number*$\}$.

Theorem 5. *For any language* L *accepted by a ROWJFA there exists a constant* k_L, *such that for every string* $w \in L$ *with* $|w| \geq k$, *there exists a permutation* $P(w)$ *of* w, *which can be written as* $P(w) = xyz$, *satisfying the following conditions:*

1. $y \neq \varepsilon$.
2. $|xy| \leq k$.
3. $xy^m z \in L$, *for all* $m \geq 0$.

Corollary 6. *There is no ROWJFA that accepts a language* $\{a^n b^n \mid n \geq 0\}$.

A similar argument works for the original JFA.

5.2 Closure properties

We study closure properties of **ROWJ** as follows.

Theorem 7. *The class* **ROWJ** *is not closed under*

1. *concatenation,*
2. *intersection,*
3. *intersection with regular languages,*

4. *concatenation with regular languages,*

5. *Kleene star,*

6. *homomorphism,*

7. *substitution,*

8. *reverse.*

It is easy to see that similar results hold for **LOWJ**. Consequently, **ROWJ** (or **LOWJ**) does not coincide with any of the following language classes: deterministic context-free languages, context-free languages, indexed languages [1], context-sensitive languages, recursive languages, or recursively enumerable languages because these are closed under intersection with regular languages.

	JFA	ROWJ
Concatenation	×	×
Union	○	?
Complement	○	?
Intersection	○	×
Intersection with REG	×	×
Kleene star	×	×
Homomorphism	×	×
ε –free Homomorphism	×	×
Substitution	×	×
Inverse Homomorphism	×	?
Reverse	○	×

○ : closure × : non–closure

Figure 6. Summary of closure properties.

5.3 Relations with well-known language families

We examine relations among **ROWJ**, **LOWJ** and some well-known language families.

Theorem 8. **ROWJ** *properly includes* **REG**.

Theorem 9. **ROWJ** $\not\subset$ **JFA**.

Theorem 10. **CF** *and* **ROWJ** *are incomparable.*

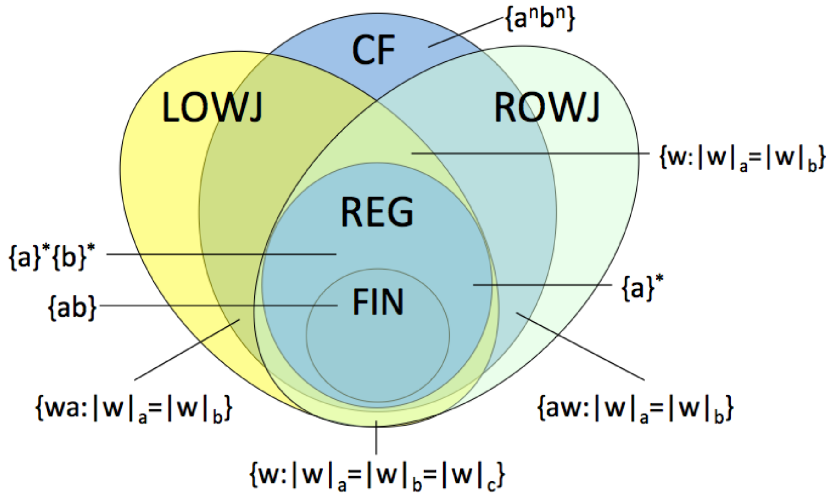

Figure 7. Relation between **ROWJ** and **LOWJ**

We compare **ROWJ** with the class defined by input revolving automata. In [2] the authors define input revolving automata and describe the way they work as follows (we recall only the description of right-revolving automata here).

Definition 11. *[2] A (nondeterministic) extended finite automaton is a 6-tuple* $A = (Q, \Sigma, \delta, \Delta, q_0, F)$*, where* Q *is a finite set of states,* Σ *is the input alphabet,* δ *and* Δ *are mappings from* $Q \times (\Sigma \cup \{\lambda\})$ *to* 2^Q *, where* δ *is called the transition function, and* Δ *is called the input operation function,* $q_0 \in Q$ *is the initial state, and* $F \subseteq Q$ *is the set of accepting states.*

The different operations on the input are formally distinguished by different interpretations of the mapping Δ. Consider configurations of extended finite automata to be tuples (q, w), where $q \in Q$ is the current state, and $w \in \Sigma^*$ is the yet unread part of the input. The transition of a configuration into a successor configuration can be induced by either δ or Δ.

- Let a be in $\Sigma \cup \{\lambda\}$ and $w \in \Sigma^*$. If p is in $\delta(q, a)$, then $(q, aw) \vdash_A (p, w)$. Those transitions are referred to as ordinary transitions.

- An input operation is performed by applying the mapping Δ. For $a \in \Sigma \cup \{\lambda\}$, $b, c \in \Sigma$, $w \in \Sigma^*$, and $p \in \Delta(q, a)$, a right-revolving transition is defined by $(q, aw) \vdash_A (p, wa)$, if $a \in \Sigma$, and $(q, bw) \vdash_A (p, wb)$ and $(q, \lambda) \vdash_A (p, \lambda)$, if $a = \lambda$.

So, right-revolving automata may skip certain letters of the input depending on the current state. This is exactly what happens when a ROWJFA reads the input word and finds a letter for which there is no transition defined from the current state.

Theorem 12. *For any ROWJFA* $M = (Q, \Sigma, R, s, F)$ *there exists a right-revolving automaton* $M' = (Q', \Sigma, \delta, \Delta, q_0, F')$, *such that* $L(M) = L(M')$.

We note that not all languages accepted by right revolving input automata are in **ROWJ**. Take, for instance, revolving automata which in every step jump over k letters regardless of what they are. Consider the right-revolving automaton $M = (\{q_0, q_1, q_a, q_b\}, \{a, b\}, \delta, \Delta, q_0, \{q_0, q_1\})$, with transitions given by $\delta(q_0, a) = q_a$, $\delta(q_0, b) = q_b$, $\delta(q_1, b) = q_b$ and $\Delta(q_a, \lambda) = q_0$, $\Delta(q_b, \lambda) = q_1$. This automaton processes the input by reading a letter and then shifting the next one to the end of the word and accepts if the letters rearranged in the order of reading them form a word from a^*b^*. It is easy to see that for a word $w \in L(M)$, if $|w|_a = |w|_b$, then $w = (ab)^n$, whereas $|w|_a = k|w|_b$ implies $w = (a^k b)^n$, etc. One can easily show, that in a word $w \in L(M)$, if the first b occurs at position i, then between every two consecutive b's there are at least $i - 1$ occurrences of a. From here, applying the techniques seen before, one gets that there exists no ROWJFA, which accepts $L(M)$.

References

[1] A. Aho, Indexed grammars - an extension of context-free grammars, *Journal of the ACM*, Vol. 15 (4), (1968) 647-671.

[2] S. Bensch, H. Bordihn, M. Holzer, and M. Kutrib. On input-revolving deterministic and nondeterministic finite automata, Information and Computation, (2009) 1140-1155.

[3] H. Chigahara, S. Fazekas, A. Yamamura, One-way Jumping Finite Automata, (in preparation).

[4] J. E. Hopcroft, and J. D. Ullman, Introduction to Automata Theory, Languages, and Computation, Addison-Wesley Publishing, Reading Massachusetts, (1979).

[5] P. Jančar, F. Mráz, M. Plátek, J. Vogel, Restarting Automata. Springer, Germany, (1995) 283-292.

[6] C. D. Manning, P. Raghavan, and H. Schütze. Introduction to Information Retrieval. Cambridge University Press, New York, (2008).

[7] A. Meduna. Automata and Languages, Theory and Applications, *Springer*, London, (2000).

[8] A. Meduna and P. Zemek. Jumping finite automata, Vol. 23, No. 7, *International Journal of Foundations of Computer Science*, (2012) 1555-1578.

[9] R. Parikh. On Context-Free Languages. Journal of the Association for Computing Machinery 13(4), (1966).